**Sitthiphol Yuwanaboon**

**John O’neal**

**CS 130**

**16 March 2021**

**PART I:      LegV8 (*30 pts.*)**

The purpose of this part of the lab is to explore (play) with the simulator in Section 2.24. There are essentially three emulators in section 2.24. The first one has an add and subtract example. The second one is giving load and store example. The third has a loop example. First, we are going to use them as they are to become familiar with the operation of the simulator. Then, we will modify them and analyze their operation or non-operation.

1. A. Step thru the first example and watch the registers and memory as each instruction executes. Write down what changes in the registers and/or memory after executing instruction. Write the changes down.
2. Changes after first instruction:

X1 =9

1. Changes after second instruction:

X4 change from 0 to 5

1. Changes after third instruction:

X7 change from 1 to 4

1. Modify the first example to add negative numbers.
2. Add -235,623,100 to 15,899,376. What is the answer in X2? Is this the right           answer? If not, what happened?

The number is subtracted with each other: the result is   
-219723724

1. Add -9223372036854775807, to -1. What is the answer in X2? Is this the

right answer? If not, what          happened?

The result is -9223372036854775808. It still add since it is still reached maximum

1. Add -9223372036854775808, to -1. What is the answer in X2? If not, what          happened?

  The result is 9223372036854775807. The sign number become positive number since memory 2^30 is reached the maximum the number can go. So the sign will change.

1. A. Follow the instructions for the second emulator (2.24.20). Paste a screenshot of         the result following the last instruction.

A picture containing graphical user interface

Description automatically generated

1. B. Find the mistakes in the following:

ADDI X10, X10, #800

ADD X2, X0, X10

ADDI X1, X0, #16

LOOP:

LDUR D0, [X1, #-16]

LDUR D2, [X1, #-8]

stall

FADDD D4, D0, D2

stall

stall

STUR D4, [X1, #0]

ADDI X1, X1, #8

CMP X1, X2

B.NE LOOP

Paste the corrected version here:

  MOV X10 #800

ADD X2, X0, X10

ADDI X1, X0, #16

LOOP: LDUR D0, [X1, #-16]

LDUR D2, [X1, #-8]

FADDD D4, D0, D2

STUR D4, [X1, #0]

ADDI X1, X1, #8

CMP X1, X2

B.LE LOOP

1. A. Follow the instructions for the third emulator (2.24.3).

1. For the first go thru (X1 =2), how many times did you go thru the loop?
2. When X1 =5, how many times thru the loop?
3. Change the B.GE to B.GT and then answer the same two questions.

                        X1 = 2, \_2\_\_times thru the loop

                        X! = 5, \_\_6\_\_\_ time thru the loop

1. B.  The following is loop that is supposed to end after 5 iterations. What’s wrong?

B.GT represent the greater than signed that it compare to 6 element because when the condition is true. It is kicked out

        Correct it and paste the corrected version here: (You can change it anyway you    like to get it to loop 5 times and 5times only.

MOV X5, #56

ADD X2, X0, X5

ADDI X1, X0, #16

LOOP:

LDUR X0, [X1, #-16]

LDUR X2, [X1, #-8]

ADD X4, X0, X2

STUR X4, [X1, #0]

ADDI X1, X1, #8

SUB XZR, X1, X2

B.LE LOOP

// less or equal is my preference rather than != because the number can be a bigger depending on loading and increment and decrement

1. Use what you have created to write a routine that finds the Fibonacci value of a number entered in X5. Put the result in X6.
2. MOV X5, #104
3. ADD X2, X0, X5
4. ADDI X1, X0, #16
5. LOOP:
6. LDUR X0, [X1, #-16]
7. LDUR X2, [X1, #-8]
8. ADD X4, X0, X2
9. STUR X4, [X1, #0]
10. ADDI X1, X1, #8
11. SUB XZR, X1, X2
12. B.LE LOOP
13. ADDI X6 X5, #0
14. Paste the code here:
15. Paste a screenshot of the result of finding the Fibonacci number for 11.

55 is Fibonacci number for 11

**PART II:    6502  (*25 pts*)**

1. Got to *Easy 6502, Instructions (*[Easy 6502 by skilldrick (Links to an external site.)](https://skilldrick.github.io/easy6502/index.html#instructions)*) .*
2. Assemble the routine and display the disassembly and hexdump windows. Take a screenshot of each and paste them in the following vertical order:

                              Emulator

Table

Description automatically generated with low confidence

disassembler

Address Hexdump Dissassembly

-------------------------------

$0600 a9 c0 LDA #$c0

$0602 aa TAX

$0603 e8 INX

$0604 69 c4 ADC #$c4

$0606 00 BRK

                              Hexdump

0600: a9 c0 aa e8 69 c4 00

1. Use Debug and Step thru the routine. What flag(s) changed after the LDA?\_N \_\_\_ after the ADC?\_\_\_\_\_C flag has changed\_\_\_ Why?

N is response to accumulator

 Carry show that there are arithmetic going on in system like add command

1. Replace the code in the emulator with the following:

:

LDA #$fc  ;Load the hex value $fc into the A register

loopstart:

ADC    #$01      ;add 1 to the value in the A register

BCS     done

BCC    loopstart

done:

BRK       ;Break - we're done

            Assemble the code, then turn on the debugger and step through the code, watching the A-        register and the carry flag. This is loop that adds 1 to the A-register every time thru the             loop.

            What happens when hex ff is added to 1?

It increment value of fc to fd

            What is in the A-register?

Fd is in the A-register

            What is the value of the carry flag?

0 since it is still not reached the maximum

            What does this mean?

 Set if an arithmetic operation generates a carry or a borrow out of the mostsignificant bit of the result; cleared otherwise

1. Use what you learned in Lab 1 and this lab to write a routine that gives you the Fibonacci value for a number loaded into the A-register at the beginning. Display the Fibonacci value in the A-register at the end.

          Use 5, 11, and 30 for  test values.

          Paste the code here.

LDY #5    ; Load required number  
 LDA #0    ; Clear last value  
 STA LAST  
 LDA #1    ; Set initial increment  
Loop:  
 TAX       ; Save current value  
 CLC       ; Compute the next one  
 ADC LAST  
 STX LAST  ; Save new last value  
 DEY  
 BNE Loop  
 NOP       ; Yth value is now in A

          Give the answers for the test values here.

                    For 5, you get \_\_\_5\_\_\_.

                    For 11, you get \_\_\_55\_\_\_.

                    For 30, you get \_\_ 317811

\_\_\_\_.